

the above application. Exhibit A is a portion of a Design Specification dated prior to January 15, 1999 which describes the design specification for implementing database tables and related structures for managing message data to be accessed by multiple recipients. Exhibit B is a copy of a driver script which invokes other SQL test scripts. Exhibit C is a copy of a script that was used to create multiple consumers queues based on the subject invention. Exhibit D are copies of scripts that perform operations such as enqueueing and dequeuing of messages, as well as cleanup of the queue tables. Exhibit E are copies of the outputs generated by the test scripts of Exhibits B-D. Exhibit F are copies of documents showing successful tests of the above scripts.

4. Sections 2.2.4 and Section 3.4 of Exhibit A describes history management processes that are implemented in a software program that was created and reduced to practice prior to January 15, 1999 which embodied the subject matter of the above application. These sections disclose history management of information for multiple consumers, where the information includes one or more information records in order from one or more queues. Section 2.2.4 describes the provision of data from information records to consumers, e.g., by de-queueing a message. This section also describes updating a history table which includes records for the consumers. Section 3.4.1 of Exhibit A describes database fields that are employed to manage history information for the processing of messages, as well as procedures for proving to users and updating such information. Specific information to track for history records are also provided in this section. The fields used to manage the history information includes at least one field that indicates whether a data item has been provided to a consumer, which is updated when a consumer accesses the information (e.g., by updating the "deq_time" field with the time at which the message was dequeued.). This database structure was implemented in a software program that was created and reduced to practice prior to January 15, 1999 which embodied the subject matter of the above application. Section 3.4.2 of Exhibit A illustrates the algorithm to update and manage index records relating to the messages and message recipients. This process was implemented in the software program that was created and reduced to practice prior to January 15, 1999 which embodied the subject matter of the above application. These sections as well as others within Exhibit A such as section 3.4.1 in the discussion regarding "a dequeue index" and in section 3.4.2 in subsection 1 and at Exhibit E, page 16 (marked as page "1" on the bottom) describe managing information to be accessed by multiple consumers,

where the information includes one or more information records, and the information records to be accessed by the multiple consumers are in a specified order, with each information record including data to be accessed by a consumer. Exhibit E, page 2 (marked as page “2” on the bottom), lines 7 and lines 35-end illustrates the successful creation of the “dequeue index” to allow information records by the multiple consumers to be accessed in a specified order.

4. The attached exhibits disclose providing data of an information record to a consumer. For example, Exhibit E, page 9 (marked as “Page 1” on bottom of page) discloses this, particularly at the “sys.message” statement on or around line 15 of the page and Exhibit E, page 10 (marked as “Page 2” on bottom of page), lines 43-end discloses this, particularly at the “Message” statements.

5. The attached exhibits disclose updating a history table, said history table comprising a history record for said consumer for said information record, said history record comprising a message state field for indicating whether said data of said information record have been provided to said consumer. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table and section 3.4.2 in subsection 3 in the discussion of “array insert...” and in Exhibit C, page 3, at or around lines 45-48, with respect to the statement “If hist(i).transaction_id IS NOT NULL...”

6. The attached exhibits disclose that said updating comprising setting said message state field in a history record corresponding to said consumer to indicate said consumer accessed said data. For example, For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table and at section 3.4.2 in the discussion regarding “2. Update deq_time....”.

7. The attached exhibits disclose that each said information record further comprises a message identifier value that identifies the data of said information record, and each said history record further comprises a message id field that identifies data in an information record. For example, Exhibit A, section 3.4.1 discloses this in subsection a with respect to the discussion of a history index for a queue table and in Exhibit D, page 3 (marked as “Page 1” on bottom of page)

discloses this, particularly at the “dbms_aq.enqueue()” statement on or around line 15 of the page relating to “enq_msgid” and at Exhibit D, page 4 (marked as “Page 2” on bottom of page) at lines 26 and 51 with references to the “dbms_aq.dequeue ()” statements.

8. The attached exhibits disclose that each said history record further comprises a consumer id field that identifies a consumer of said multiple consumers that is to access data in an information record, said data identified by said message id field in said history record, said consumer id field of said history record identifying said history record as corresponding to said consumer. For example, Exhibit A, section 3.4.1 discloses this in subsection e with respect to the discussion of a history index for a queue table and in section 3.4.2 in subsection 3 in the discussion of “array insert...” and in section 3.4.2 in the discussion regarding “2. Update deq_time...” with respect to “r1” and at Exhibit D, page 4 (marked as “Page 2” on bottom of page) at lines 21 and 47 with references to the “deqopt.consumer_name := consumer;” statements, Exhibit E, page 13 (marked as “Page 1” on bottom of page) regarding the “dequeue_options.consumer_name := subscriber” statement.

9. The attached exhibits disclose that updating comprises setting said message state field in the history record with a message id field that identifies said data that said consumer is provided access to and with a consumer id field that identifies said consumer. For example, Exhibit A, section 3.4.1 discloses this in subsection e with respect to the discussion of a history index for a queue table and in section 3.4.2 in subsection 3 in the discussion of “array insert...” and in section 3.4.2 in the discussion regarding “2. Update deq_time...” with respect to “r1” and at Exhibit D, page 4 (marked as “Page 2” on bottom of page) at lines 21 and 47 with references to the “deqopt.consumer_name := consumer;” statements Exhibit E, page 13 (marked as “Page 1” on bottom of page) regarding the “dequeue_options.consumer_name := subscriber” statement.

10. The attached exhibits disclose storing data to be accessed by a consumer in an information record. For example, Exhibit D, page 3 (marked as “Page 1” on bottom of page) discloses this, particularly at the “dbms_aq.enqueue()” statement on or around line 15 of the page relating to “enq_userdata” and at Exhibit E page 8 (marked as “Page 1” on bottom of page), line 18 regarding the statement “l1>...” With respect to the “enq_userdata” element.

11. The attached exhibits disclose creating a history record for each consumer that is to access said data. For example, Exhibit A, section 3.4.1 discloses this in subsection e with respect to the discussion of a history index for a queue table and in section 3.4.2 in subsection 3 in the discussion of “array insert...”.

12. The attached exhibits disclose setting said message state field in each said history record to indicate said data has not been accessed. For example, Exhibit A, section 3.4.2 discloses this in subsection 3 in the discussion of “array insert...” with respect to the “NULL” elements.

13. The attached exhibits disclose identifying the data of an information record that a consumer is to be provided access to by order data in a read-order table, said order data indicating a relative order that data in said information records is to be accessed by said multiple consumers. For example, Exhibit A, section 3.4.1 discloses this in the discussion regarding “a dequeue index” and in section 3.4.2 in subsection 1 and at Exhibit E, page 16 (marked as page “1” on the bottom) and at Exhibit E, page 2 (marked as page “2” on the bottom), lines 7 and lines 35-end.

14. The attached exhibits disclose reading one or more history records of said history table, said one or more history records comprising a history table read. For example, Exhibit A, section 3.4.1 discloses this with respect to the discussion of a history index for a queue table.

15. The attached exhibits disclose deleting an information record if all the message state fields in all of the history records of said history table read indicate that said data in said information record has been accessed. For example, this is disclosed in Exhibit A, section 3.4.2, last paragraph, regarding removal of messages.

16. The attached exhibits disclose associating a work list table with said history table, said work list table comprising one or more work entries, each said work entry comprising an identification of data in an information record. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

17. The attached exhibits disclose adding a work entry to said work list table, said work entry comprising an identification of said data said consumer is provided access to. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

18. The attached exhibits disclose accessing a work entry in said work list table. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

19. The attached exhibits disclose reading one or more history records of said history table, said one or more history records comprising a history table read, said one or more history records comprising said history table read determined by said work entry. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

20. The attached exhibits disclose deleting an information record if all the message state fields in all of the history records of said history table read indicate that said data in said information record has been accessed. For example, this is disclosed in Exhibit A, section 3.4.2, last paragraph, regarding removal of messages.

21. The attached exhibits disclose results from a system and method that implements batching two or more work entries in said work list table. For example, the disclosure of Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities was implemented with batching for the test system that produced the results shown in Exhibits E and F.

22. The attached exhibits disclose reading one or more history records of said history table, said one or more history records determined by said two or more work entries. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

23. The attached exhibits disclose deleting one or more information records. For example, this is disclosed in Exhibit A, section 3.4.2, last paragraph, regarding removal of messages.

24. The attached exhibits disclose an information queue comprising one or more information queue records. For example, this is disclosed in Exhibit A, section 3.4.1 regarding a queue table.

25. The attached exhibits disclose each said information queue record comprising information to be accessed by one or more consumers. For example, this is disclosed in Exhibit A, section 3.4.1 regarding a queue table.

26. The attached exhibits disclose a table separated from said information queue, said table comprising one or more table records, each said table record comprising an identification of said information in an information queue record, each said table record further comprising a consumer identification field comprising an identification of one of said one or more consumers, and a message state field for indicating whether one of the one or more information queue records has been accessed by one of the one or more consumers. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table.

27. The attached exhibits disclose that each information queue record comprises identification of information of an information queue record. For example, Exhibit A, section 3.4.1 discloses this in subsection a with respect to the discussion of a history index for a queue table and in Exhibit D, page 3 (marked as "Page 1" on bottom of page) discloses this, particularly at the "dbms_aq.enqueue()" statement on or around line 15 of the page relating to "enq_msgid".

28. The attached exhibits disclose a read-order table, said read-order table comprising order data indicating the order that information in said information queue is to be delivered to a consumer. For example, this is disclosed in Exhibit A, section 3.4.2 regarding the discussion of "1. insert one key...".

29. The attached exhibits disclose that a read-order table comprises one or more records, each said record of said read-order table comprising an identification field that identifies

information in an information queue record, each said record of said read-order table further comprising an enqueue time field that comprises said order data. For example, this is disclosed in Exhibit A, section 3.4.2, regarding the discussion of “1. insert one key...” and regarding “1. Delete its index entry from qt_i.”

30. The attached exhibits disclose a work list table, said work list table comprising one or more work list entries, each said work list entry comprising an identification of information in an information queue record. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

31. The attached exhibits disclose that a work list entry is a record. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

32. The attached exhibits disclose that a work list table comprises one or more work records and each said work list entry is a field in a work record. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

33. The attached exhibits disclose a message queue comprising one or more message queue records, each said one or more message queue records comprising a message and a message identification. For example, this is disclosed in Exhibit A, section 3.4.1 regarding a queue table.

34. The attached exhibits disclose a history table separated from said message queue comprising one or more history records, each of said one or more history records comprising a message identification, a consumer identification and a message state identification, each said message state identification indicating whether one of the one or more message queue records has been accessed. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table.

35. The attached exhibits disclose a work list table separated from said message queue and said history table comprising one or more work list entries, each said work list entry comprising a message identification. For example, this is disclosed in Exhibit A, section 3.4.2, in the discussion relating to a time manager index and the list of time-management activities.

36. The attached exhibits disclose a method for multiple consumers to access information in a non first-in first-out, prescribed order, said information comprising one or more pieces of information, a first piece of information stored in a first location. For example, this is disclosed in Exhibit C, page 2, line 1-10, particularly with respect to the "sort_list" element(s) of the statement(s).

37. The attached exhibits disclose providing access to first piece of information to a first consumer of said multiple consumers; indicating in a second location in a history table that said first consumer has accessed said first piece of information, said history table having a first message state field for indicating whether said first consumer has accessed said first piece of information; providing access to said first piece of information to a second consumer of said multiple consumers; and indicating in a third location in said history table that said second consumer has accessed said first piece of information, said history table having a second message state field for indicating whether said second consumer has accessed said first piece of information. For example, this is disclosed in Exhibit E, page 11, lines 25-end and at page 17, lines 1-24.

38. The attached exhibits disclose that a first location comprises an information entry in a queue of information. For example, this is disclosed in Exhibit E, pages 1-3.

39. The attached exhibits disclose a queue of information comprises one or more information entries, and each said information entry comprises a piece of information to be accessed by one or more of said multiple consumers, each said information entry further comprising an identification of said piece of information in said information entry. For example, this is disclosed in Exhibit E, pages 1-3.

40. The attached exhibits disclose deleting an entry comprising a first piece of information that a first consumer and a second consumer is provided access to from said queue of information after said first consumer and said second consumer have accessed said first piece of information. For example, this is disclosed in Exhibit A, section 3.4.2, last paragraph, regarding removal of messages.

41. The attached exhibits disclose for each of one or more consumers, a table comprises a separate table record for each piece of information to be accessed by said consumer. For example, this is disclosed in Exhibit A, section 3.4.1 and section 3.4.2.

42. The attached exhibits disclose the history table comprises an identification of a first piece of information and an identification of a first consumer. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table.

43. The attached exhibits disclose a history entry in said history table, said history entry comprising an identification of a first piece of information and an identification of a second consumer. For example, Exhibit A, section 3.4.1 discloses this in subsections a-k with respect to the discussion of a history index for a queue table.

44. The attached exhibits disclose indicating in a location an order in which one or more pieces of information is to be accessed by multiple consumers. For example, this is disclosed in Exhibit C, page 2, line 1-10, particularly with respect to the "sort_list" element(s) of the statement(s).

45. The subject invention was reduced to practice and tested to verify that it works for its intended purpose prior to January 15, 1999. This is shown by Exhibits B-F which include copies of documents which evidence that the subject invention was tested and found to work for its intended purpose. As noted above, Exhibit B is a copy of a driver script which invokes the other SQL test scripts. Exhibit C is a copy of a script that was used to create multiple consumers queues based on the subject invention. Exhibit D are copies of scripts that perform operations such

as enqueueing and dequeuing of messages, as well as cleanup of the queue tables. Exhibit E are copies of the outputs generated by the test scripts of Exhibits B-D. Exhibit F show copies of documents showing successful tests of the above scripts. Regression tests were performed to verify that the operations performed by the test scripts generate the specified output. The documents in Exhibit F are copies of files that display the results of running the tests and demonstrate that the tests ran correctly with the expected results.

13. We further declare that all statements made herein of my own knowledge are true and all statements made on information and belief are believed to be true; and further that these statements were made with the knowledge that willful false statements and the like so made are punishable by fine or imprisonment, or both, under Section 1001 of Title 18 of the United States Code and that such willful false statements may jeopardize the validity of the application or any patent issued thereon.

Date: June 21 2007



Sashikanth Chandrasekaran

Patent
237/116
OI7011472001

Date: June 22, 2007

Ashok Saxena

Ashok Saxena

EXHIBIT A

Design Specification for AQ Propagation , RDBMS, 8.1

Project ID:aq_propagation

Version: [REDACTED]

Status: Approved

Author: Sashi Chandrasekaran, Ashok Saxena

[REDACTED]
[REDACTED]
[REDACTED]

Version	Reviewers	Changes
[REDACTED]	arsaxena	Creation
[REDACTED]	[REDACTED] [REDACTED] [REDACTED] [REDACTED] [REDACTED] [REDACTED] [REDACTED]	[REDACTED]

[REDACTED]
[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

2.2.4 **History Management**

History management for multi-consumer queues leaves a lot to be desired. There are two fundamental problems to history management: storage and reference counting.

- **Storage**

The history information is stored as a varray object collection. The current interface to varray collections retrieves and unpickles the entire collection. AQ uses C interfaces to navigate through the collection and update the history element for the appropriate consumer. The updated collection is written entirely into the database for history tracking.

- **Reference Counting**

[illegible]

~~CONFIDENTIAL~~

[REDACTED]

3.4 History Management

3.4.1 Data Structures

When a queue table is created, three additional IOTs are created to store the message meta data. The message data and message properties are kept in the queue table. The queue table in 8.1 will be modified for changes in message properties, but they are not relevant to the history management. The three additional IOTs are:

- A dequeue index to maintain the sort-order of messages for each recipient. This index will have the same structure as in Oracle 8.0.
- A history index that maintains the history of processing of every message. The columns in this IOT are as follows:

- a. msgid - unique identifier of the message
- b. rowid - location of the message in the queue table.
- c. address - address of the recipient.

It is the source queue name (without the schema name appended to it) if the consumer will dequeue messages directly from the source queue. The address supported by AQ propagation will be of the form [schema.]queue[@database_link]. Messages are propagated to the destination queue specified by the address. AQ does not require global names be set to TRUE, however it is recommended. The database link name is resolved in the context of the owner of the source queue.

- d. protocol - protocol field of the recipient structure.

This field qualifies the address. It is the session-level protocol (e.g. dblink/TIB) used to propagate messages to the destination queue. It is 0 if the address is a database link address or if the consumer dequeues the message from the local queue.

- e. consumer_name - name of the agent (recipient) that dequeued the message.
- f. txn_id - transaction id of the dequeuing transaction.
- g. deq_time - time of dequeue.
- h. deq_user - database schema id of dequeuer.
- j. propagated_msgid - message id of the enqueued message in the destination queue.

This is NULL if the address is NULL.

- k. retry_count - # times message was dequeued in remove mode (and aborted).

Columns a, c, d and e form the primary key. We may choose to include the other columns also as part of the primary key to simplify access to these columns (Key columns are easier to extract than non-key columns and also do not have the complexity of an overflow segment). Key-compression will not be used since we do not expect the prefix (msgid) to be repeated often.

- A time-manager index that maintains the list of time-management activities. The time-manager index has four columns:

- a. time - absolute time at which time-manager has to perform an operation.

- b. msgid - message id of message that needs to be acted upon.
- c. action - a description of the action that needs to be performed. The possible values are:
 1. MAKE_READY - make message available for dequeue to consumers after the delay time has passed.
 2. EXPIRE - move message to exception queue if message has not yet been processed.
 3. REMOVE - remove message after the retention time has passed.
- d. transaction_id. This is the transaction_id of the transaction that inserts the time-management entry. This is needed to generate a unique key, since two consumers can dequeue the same message and post the time-manager to perform an action at the same time. This is set if the action is REMOVE.

Columns a, b and d form the primary key. This IOT is similar to the time-manager index for Oracle 8.0 queue tables. The differences are:

- a. The IOT stores the msgid of the message rather than its rowid.
- b. There is an action column to help the time-manager determine what time-management activity needs to be performed on the message. In theory, this column is superfluous because the time-manager can deduce what action needs to be performed based on the history information in the history table. Oracle 8.0's time-manager index deduced what action needs to be performed based on the state of the message in the queue table.
- c. There could be multiple rows for the same message in the index. In fact, there could be up to one row for each agent that dequeues the message from the queue table. This is because each agent that dequeues messages independently notifies the time-manager without knowledge of the state of the message with respect to other recipients.

3.4.2 Design Description

We illustrate the use of these index structures using a simple example. Let us assume that a queue table, say qt, has been created. Call the dequeue sort order index qt_i, the history index as qt_h and the time-manager index as qt_t. Let us say a message is enqueued in queue q with the following properties: messageid = m, delay = d, expiration = e, retention time = r, recipients = {r1, r2@boston} where r1 is a local consumer and boston is a remote database. The acknowledgment mode for this message is assumed to be ACK_DEQUEUED (a propagator and dequeuer perform similar actions if the acknowledgment mode is ACK_PROPAGATED or NO_ACK).

When the message is enqueued at rowid = rid, the index structures are updated as follows:

1. insert one key into qt_i for the propagator. This step is identical to Oracle 8.0. This step is necessary so that the propagator can dequeue the message without waiting for the delay time.
2. if d is non-NULL insert key into qt_t with value [d, m, MAKE_READY, txnid] else if e is non-NULL insert key into qt_t with value [e, m, EXPIRE, txnid]
3. array insert two keys into qt_h with values [m, rid, r2@boston, 0, r2, NULL, NULL, NULL, NULL, 0] and [m, rid, q, 0, r1, NULL, NULL, NULL, NULL, 0]. This step will substitute generating the history collection in an 8.0 queue table.

When the delay time has passed the time-manager performs the following actions.

1. for each entry in qt_h where msgid = m and address = q and txn_id = txnid insert key into qt_i to enable consumer to dequeue message.
2. update the qt_t key to [e, m, EXPIRE, cur_txnid] if e is non-NULL.

Agent r1 performs the following steps after dequeuing message m.

1. Delete its index entry from qt_i.

2. Update deq_time, deq_user, txn_id columns in qt_h for row with consumer_name = r1.
3. If retention_time is not NULL, insert key [r, m, REMOVE, cur_txnid] into qt_t else if queue has no retention, insert key [gettimeofday(),m, REMOVE, cur_txnid].

The propagator updates the propagated_msgid column in qt_h and deletes the index entry from qt_i as soon as m is successfully propagated to boston. The deq_time column in qt_h and time_manager index qt_t are updated only on receipt of acknowledgment from boston that r2 has processed the message.

The time-manager marks the message as expired at time e, if either r1's or r2's deq_time columns in qt_h is NULL. Likewise, it removes the message m at time r only if r1 and r2 deq_time columns are non-NULL. In all cases the time-manager removes the index entry from the time-manager-index when it processes the entry, regardless of whether processing it resulted in any state change or not. When a message expires, the history keys in qt_h are copied over to a different queue table if the exception queue resides in a different queue table. The history keys are deleted along with the message itself when the application uses dequeue-by-message-id to remove the message from the exception queue.

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

[REDACTED]

EXHIBIT B

```

# [REDACTED]
# [REDACTED]
# tkaqnsbd.tsc
# [REDACTED]
#
# NAME
#   tkaqnsbd.tsc - Advanced Queueing New-style Short Basic Dequeue test
#
# DESCRIPTION
#   short regression test of 8.1 style multiconsumer queues
#
# NOTES
#   tests creation, navigation, enqueue, dequeue
#   [REDACTED]
#   [REDACTED]
#
# RUNS_STANDALONE Yes
# TEST_TYPE       short
# USES            SQL
# MAX_USERS       1
#
# open_cursors, shared_pool_size settings are temporary workarounds
rdbmsini O7_DICTIONARY_ACCESSIBILITY=true compatible=8.1.0 db_block_buffers=400
open_cursors=500 shared_pool_size=8000000

get tkaqnmin.sql
sql tkaqnmin > tkaqnsb1
compare tkaqnsb1 tkaqnmin mask

# Enqueue some messages and navigate thru the queues
get tkaqmnnav.sql
sql tkaqmnnav > tkaqnsb2
compare tkaqnsb2 tkaqmnnav mask

# Dequeue some messages when there is more than one message for an application
# First create the messages in the queue.
get tkaqmbdm.sql
sql tkaqmbdm > tkaqnsb3
compare tkaqnsb3 tkaqmbdm mask

get tkaqnsd1.sql
sql tkaqnsd1 > tkaqnsb4
compare tkaqnsb4 mask

# Cleanup
get tkaqmc1n.sql
sql tkaqmc1n > tkaqnsb5
compare tkaqnsb5 mask

shutdown

```

EXHIBIT C


```

# Create a queue with priority and enq_time as the sort order
execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtpeqt',
queue_payload_type => 'message', sort_list => 'priority,enq_time', multiple_consumers
=> true, comment => 'Creating queue with priority and enq_time sort order',
compatible => '8.1.3');

# Create a queue with priority as the sort order
execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtprpri',
queue_payload_type => 'message', sort_list => 'priority', multiple_consumers =>
true, comment => 'Creating queue with priority sort order', compatible => '8.1.3');

Rem      CHECK IF ALL'S OK

select schema, name, flags objno from system.aq$queue_tables;
select orderbypos, colno, name, sort_order, table_objno from
system.aq$queue_table_sort order by table_objno, orderbypos;

Rem Ensure that the tables and the indices have been created

select count(*) from tkaqqtdef;
select count(*) from aq$tkaqqtdef_i;

select count(*) from tkaqqtpeqt;
select count(*) from aq$tkaqqtpeqt_i;

select count(*) from tkaqqtprpri;
select count(*) from aq$tkaqqtprpri_i;

Rem create two queues in each queue table

Rem create two queues in tkaqqtdef
execute dbms_aqadm.create_queue(queue_name => 'q1def', queue_table =>
'sys.tkaqqtdef', max_retries => 2, comment => 'queue 1 in tkaqqtdef');
execute dbms_aqadm.create_queue(queue_name => 'q2def', queue_table =>
'sys.tkaqqtdef', max_retries => 2, comment => 'queue 2 in tkaqqtdef');

Rem create two queues in tkaqqtpeqt
execute dbms_aqadm.create_queue(queue_name => 'q1peqt', queue_table =>
'sys.tkaqqtpeqt', max_retries => 2, comment => 'queue 1 in tkaqqtpeqt');
execute dbms_aqadm.create_queue(queue_name => 'q2peqt', queue_table =>
'sys.tkaqqtpeqt', max_retries => 2, comment => 'queue 2 in tkaqqtpeqt');

Rem create two queues in tkaqqtprpri
execute dbms_aqadm.create_queue(queue_name => 'q1prpri', queue_table =>
'sys.tkaqqtprpri', max_retries => 2, comment => 'queue 1 in tkaqqtprpri');
execute dbms_aqadm.create_queue(queue_name => 'q2prpri', queue_table =>
'sys.tkaqqtprpri', max_retries => 2, comment => 'queue 2 in tkaqqtprpri');

Rem create an exception queue in the tkaqqtdef table
execute dbms_aqadm.create_queue(queue_name => 'exceptionq', queue_table =>
'sys.tkaqqtdef', queue_type => DBMS_AQADM.EXCEPTION_QUEUE, comment => 'exception q
in tkaqqtdef');

```

Rem Create procedure to check the list of subscribers for each queue

```
CREATE OR REPLACE PROCEDURE TKAQ_SUBSCRIBERS(qname VARCHAR2) AS
```

```
subs    dbms_aqadm.aq$subscriber_list_t;
nsubs   BINARY_INTEGER;
i       BINARY_INTEGER;

begin

    subs := dbms_aqadm.queue_subscribers(qname);

    dbms_output.put_line(qname);
    dbms_output.put_line('-----');

    nsubs := subs.COUNT;
    FOR i IN 0..nsubs-1 LOOP
        IF subs(i) IS NOT NULL THEN
            dbms_output.put_line('--> ' || subs(i).name);
        END IF;
    END LOOP;
```

```
end TKAQ_SUBSCRIBERS;
```

```
CREATE OR REPLACE PROCEDURE TKAQ_HISTORY(enqmsgid RAW) AS
```

```
hist          sys.aq$history;
nsubs         BINARY_INTEGER;
i             BINARY_INTEGER;

begin

    select history into hist
    from   tkagqtdf
    where  msgid = enqmsgid;

    dbms_output.put_line('-----');
    IF hist IS NOT NULL THEN
        nsubs := hist.COUNT;
        dbms_output.put_line(nsubs);
        FOR i IN 1..nsubs LOOP
            IF hist(i) IS NOT NULL THEN
                dbms_output.put_line('APP --> ' || hist(i).consumer);
                IF hist(i).transaction_id IS NOT NULL THEN
                    dbms_output.put_line('TXN --> ' || hist(i).transaction_id);
                    dbms_output.put_line('DEQ_USER --> ' || hist(i).deq_user);
                ELSE
                    dbms_output.put_line('TXN --> ');
                END IF;
            END IF;
        END LOOP;
    END IF;
    dbms_output.put_line('-----');
```

```
end TKAQ_HISTORY;
```

Rem check subscribers for queues created

set serveroutput on

```
execute tkaq_subscribers('Q1DEF');
execute tkaq_subscribers('Q2DEF');
execute tkaq_subscribers('Q1PEQT');
execute tkaq_subscribers('Q2PEQT');
execute tkaq_subscribers('Q1PRI');
execute tkaq_subscribers('Q2PRI');
```

Rem add some default subscribers for each queue.

```
declare
appl_qldef sys.aq$_agent;
begin
appl_qldef := sys.aq$_agent('appl_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
end;
/
```

```
declare
appl_qldef sys.aq$_agent;
begin
appl_qldef := sys.aq$_agent('app2_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
end;
/
```

```
declare
appl_qldef sys.aq$_agent;
begin
appl_qldef := sys.aq$_agent('app3_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
end;
/
```

```
declare
appl_qldef sys.aq$_agent;
begin
appl_qldef := sys.aq$_agent('app4_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
end;
/
```

```
declare
appl_qldef sys.aq$_agent;
begin
appl_qldef := sys.aq$_agent('app5_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
end;
/
```

```
declare
appl_qldef sys.aq$_agent;
begin
appl_qldef := sys.aq$_agent('app6_qldef', NULL, NULL);
dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
end;
/
```

```
declare
appl_qldef sys.aq$_agent;
begin
appl_qldef := sys.aq$_agent('appl_q2def', NULL, NULL);
```

```

dbms_aqadm.add_subscriber('sys.q2def',app1_q1def);
end;
/

declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_agent('app2_q2def', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2def',app1_q1def);
end;
/

declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_agent('app1_q1peqt', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q1peqt',app1_q1def);
end;
/

declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_agent('app2_q1peqt', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q1peqt',app1_q1def);
end;
/


declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_agent('app1_q2peqt', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2peqt',app1_q1def);
end;
/

declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_agent('app2_q2peqt', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2peqt',app1_q1def);
end;
/

declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_agent('app1_q1pri', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q1pri',app1_q1def);
end;
/

declare
app1_q1def sys.aq$_agent;
begin
app1_q1def := sys.aq$_agent('app2_q1pri', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q1pri',app1_q1def);
end;
/

```



```
declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_agent('app1_q2pri', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2pri',app1_qldef);
end;
/
```

```
declare
app1_qldef sys.aq$_agent;
begin
app1_qldef := sys.aq$_agent('app2_q2pri', NULL, NULL);
dbms_aqadm.add_subscriber('sys.q2pri',app1_qldef);
end;
/
```

Rem check subscribers for queues created

```
execute tkaq_subscribers('Q1DEF');
execute tkaq_subscribers('Q2DEF');
execute tkaq_subscribers('Q1PEQT');
execute tkaq_subscribers('Q2PEQT');
execute tkaq_subscribers('Q1PRI');
execute tkaq_subscribers('Q2PRI');
```

Rem start the queues

```
execute dbms_aqadm.start_queue(queue_name => 'sys.q1def');
execute dbms_aqadm.start_queue(queue_name => 'sys.q2def');
execute dbms_aqadm.start_queue(queue_name => 'sys.q1peqt');
execute dbms_aqadm.start_queue(queue_name => 'sys.q2peqt');
execute dbms_aqadm.start_queue(queue_name => 'sys.q1pri');
execute dbms_aqadm.start_queue(queue_name => 'sys.q2pri');
```

EXHIBIT D


```
Rem
Rem $Header: tkaqmc1n.sql 27-jul-97.13:41:57 nbhatt Exp $
Rem
Rem tkaqmc1n.sql
Rem
Rem
Rem
Rem NAME
Rem     tkaqmc1n.sql - <one-line expansion of the name>
Rem
Rem DESCRIPTION
Rem     <short description of component this file declares/defines>
Rem
Rem NOTES
Rem     <other useful comments, qualifications, etc.>
Rem
Rem
Rem
Rem
Rem
Rem
Rem
Rem
```

```
Rem TODO: The script must execute as tkaqadm
Rem connect tkaqadm/tkaqadm
```

```
set echo on
```

```
connect sys/knl_test7 as sysdba
```

```
Rem cleanup the queue tables
```

```
execute dbms_aqadm.drop_queue_table('sys.tkaqqtdef', TRUE);
execute dbms_aqadm.drop_queue_table('sys.tkaqqtpegt', TRUE);
execute dbms_aqadm.drop_queue_table('sys.tkaqqtpr', TRUE);
```

```
Rem connect as sys and drop the types
connect sys/knl_test7 as sysdba
```

```
drop type message;
```

```
Rem disable aq logins
```

```
drop user tkaquser cascade;
drop user tkaqadm cascade;
```



```

Rem [REDACTED]
Rem [REDACTED]
Rem [REDACTED]
Rem tkaqmnnav.sql
Rem [REDACTED]
Rem [REDACTED]
Rem NAME
Rem     tkaqmnnav.sql - <one-line expansion of the name>
Rem
Rem DESCRIPTION
Rem     <short description of component this file declares/defines>
Rem
Rem NOTES
Rem     <other useful comments, qualifications, etc.>
Rem [REDACTED]
Rem [REDACTED]
Rem [REDACTED]
Rem [REDACTED]
Rem [REDACTED]
Rem [REDACTED]

```

```

set echo on
connect tkaquser/tkaquser
set serveroutput on

```

```

create or replace procedure tkaq_navenq(priority in number) as
enq_userdata sys.message;
enq_msgid     raw(16);
enqopt        dbms_aq.enqueue_options_t;
msgprop       dbms_aq.message_properties_t;

```

```
begin
```

```

enq_userdata := sys.message(priority, 'HELLO, WORLD!');
msgprop.priority := priority;
dbms_aq.enqueue('sys.q2pri', enqopt, msgprop, enq_userdata, enq_msgid);
end;
/

```

```

execute tkaq_navenq(1);
execute tkaq_navenq(2);
commit;

```

```

execute tkaq_navenq(3);
execute tkaq_navenq(4);
commit;

```

```

execute tkaq_navenq(5);
execute tkaq_navenq(6);
commit;

```

```

execute tkaq_navenq(7);
execute tkaq_navenq(8);
commit;

```

```

execute tkaq_navenq(9);
execute tkaq_navenq(10);
commit;

```

```

execute tkaq_navenq(11);
execute tkaq_navenq(12);

```

```

commit;

execute tkaq_navenq(13);
execute tkaq_navenq(14);
commit;

execute tkaq_navenq(15);
execute tkaq_navenq(16);
commit;

create or replace procedure tkaq_navdeq(consumer IN VARCHAR2) as
deq_userdata sys.message;
deq_msgid      raw(16);
deqopt         dbms_aq.dequeue_options_t;
msgprop        dbms_aq.message_properties_t;

begin

deqopt.wait := DBMS_AQ.NO_WAIT;
deqopt.consumer_name := consumer;
deqopt.navigation := DBMS_AQ.FIRST_MESSAGE;
deqopt.dequeue_mode := DBMS_AQ.BROWSE;

FOR i in 1..9 loop
    dbms_aq.dequeue('sys.q2pri', deqopt, msgprop, deq_userdata, deq_msgid);
    dbms_output.put_line('Message: ' || deq_userdata.id
                        || ':' || deq_userdata.data);
    commit;
    deqopt.navigation := DBMS_AQ.NEXT_MESSAGE;
END LOOP;

end;
/

execute tkaq_navdeq('app1_q2pri');
execute tkaq_navdeq('app2_q2pri');

create or replace procedure tkaq_navdeq(consumer IN VARCHAR2) as
deq_userdata sys.message;
deq_msgid      raw(16);
deqopt         dbms_aq.dequeue_options_t;
msgprop        dbms_aq.message_properties_t;

begin

deqopt.wait := DBMS_AQ.NO_WAIT;
deqopt.consumer_name := consumer;
deqopt.navigation := DBMS_AQ.FIRST_MESSAGE;

FOR i in 1..19 loop
    dbms_aq.dequeue('sys.q2pri', deqopt, msgprop, deq_userdata, deq_msgid);
    dbms_output.put_line('Message: ' || deq_userdata.id
                        || ':' || deq_userdata.data);
    commit;
    deqopt.navigation := DBMS_AQ.NEXT_MESSAGE;
END LOOP;

end;
/

execute tkaq_navdeq('app1_q2pri');
execute tkaq_navdeq('app2_q2pri');

```



```


[REDACTED]
dequeue_options.consumer_name := subscriber;
dequeue_options.navigation := DBMS_AQ.FIRST_MESSAGE;
dequeue_options.wait := 1;
dbms_aq.dequeue(queue_name=> 'sys.q2def',
                 dequeue_options=>dequeue_options,
                 message_properties=>message_properties,
                 payload=>deq_userdata,
                 msgid=>deq_msgid);

commit;

dbms_output.put_line('MSG-> ' || deq_userdata.id || ' ' || deq_userdata.data);
end;
/

Rem enqueue twelve messages each subscriber should get two.
execute tkaq_mbasicenq(1, 'First Message');
execute tkaq_mbasicenq(2, 'Second Message');
execute tkaq_mbasicenq(3, 'Third Message');
execute tkaq_mbasicenq(4, 'Fourth Message');
execute tkaq_mbasicenq(5, 'Fifth Message');
execute tkaq_mbasicenq(6, 'Sixth Message');
execute tkaq_mbasicenq(7, 'Seventh Message');
execute tkaq_mbasicenq(8, 'Eight Message');
execute tkaq_mbasicenq(9, 'Ninth Message');
execute tkaq_mbasicenq(10, 'Tenth Message');
execute tkaq_mbasicenq(11, 'Eleventh Message');
execute tkaq_mbasicenq(12, 'Twelveth Message');
execute tkaq_mbasicenq(13, 'Thirteenth Message');
execute tkaq_mbasicenq(14, 'Fourteenth Message');
execute tkaq_mbasicenq(15, 'Fifteenth Message');
execute tkaq_mbasicenq(16, 'Sixteenth Message');
execute tkaq_mbasicenq(17, 'Seventeenth Message');
execute tkaq_mbasicenq(18, 'Eighteenth Message');
execute tkaq_mbasicenq(19, 'Nineteenth Message');
execute tkaq_mbasicenq(20, 'Twentyth Message');
execute tkaq_mbasicenq(21, 'Twentyfirst Message');
execute tkaq_mbasicenq(22, 'Twenty2nd Message');
execute tkaq_mbasicenq(23, 'Twenty3rd Message');

```



```
execute tkaq_mbasicenq(24, 'Twenty4th Message');  
commit;
```

EXHIBIT E

```

Echo
SVRMGR> connect sys/knl_test7 as sysdba
Connected.
SVRMGR>
SVRMGR>
SVRMGR> Rem
SVRMGR> Rem Create a queue user and administrator
SVRMGR>
SVRMGR> grant connect,resource,aq_administrator_role to tkaqadm identified by
tkaqadm;
Statement processed.
SVRMGR>
SVRMGR> grant connect,resource,aq_user_role to tkaquser identified by tkaquser;
Statement processed.
SVRMGR>
SVRMGR> grant execute on dbms_aq to tkaquser;
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('ENQUEUE_ANY','tkaquser',FALSE);
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('DEQUEUE_ANY','tkaquser',FALSE);
Statement processed.
SVRMGR> grant execute on dbms_aqadm to tkaquser;
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('MANAGE_ANY','tkaquser',TRUE);
Statement processed.
SVRMGR> grant execute on dbms_aq to tkaqadm;
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('ENQUEUE_ANY','tkaqadm',FALSE);
Statement processed.
SVRMGR> execute dbms_aqadm.grant_system_privilege('DEQUEUE_ANY','tkaqadm',FALSE);
Statement processed.
SVRMGR> execute dbms_aqadm.grant_type_access('tkaqadm');
Statement processed.
SVRMGR>
SVRMGR> Rem
SVRMGR> Rem Create a type
SVRMGR>
SVRMGR> create type message as object(id NUMBER, data VARCHAR2(30));
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> grant execute on message to tkaqadm;
Statement processed.
SVRMGR> grant execute on message to tkaquser;
Statement processed.
SVRMGR>
SVRMGR> Rem TODO: the remaining SQL scripts must be executed as tkaqadm
SVRMGR> Rem connect tkaqadm/tkaqadm
SVRMGR>
SVRMGR> # Create a queue with default sort ordering
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtdef',
queue_payload_type => 'message', multiple_consumers => true, comment => 'Creating
queue table with default sort ordering', compatible => '8.1.3');
Statement processed.
SVRMGR>
SVRMGR> # Create a queue with priority and enq_time as the sort order
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtpeqt',
queue_payload_type => 'message', sort_list => 'priority,enq_time', multiple_consumers
=> true, comment => 'Creating queue with priority and enq_time sort order',
compatible => '8.1.3');

```

Statement processed.

SVRMGR>

SVRMGR> # Create a queue with priority as the sort order

SVRMGR>

SVRMGR> execute dbms_aqadm.create_queue_table(queue_table => 'sys.tkaqqtprl',
queue_payload_type => 'message', sort_list => 'priority', multiple_consumers =>
true, comment => 'Creating queue with priority sort order', compatible => '8.1.3');
Statement processed.

SVRMGR>

SVRMGR> Rem CHECK IF ALL'S OK

SVRMGR>

SVRMGR> select schema, name, flags objno from system.aq\$_queue_tables;

SCHEMA	NAME	OBJNO
SYSTEM	DEF\$_AQCALL	0
SYSTEM	DEF\$_AQERROR	0
SYS	TKAQQTDEF	9
SYS	TKAQQTPEQT	9
SYS	TKAQQTPRI	9

5 rows selected.

SVRMGR> select orderbypos, colno, name, sort_order, table_objno from
system.aq\$_queue_table_sort order by table_objno, orderbypos;
select orderbypos, colno, name, sort_order, table_objno from
system.aq\$_queue_table_sort order by table_objno, orderbypos

*

ORA-00942: table or view does not exist

SVRMGR>

SVRMGR> Rem Ensure that the tables and the indices have been created

SVRMGR>

SVRMGR> select count(*) from tkaqqtdef;

COUNT(*)

0

1 row selected.

SVRMGR> select count(*) from aq\$_tkaqqtdef_i;

COUNT(*)

0

1 row selected.

SVRMGR>

SVRMGR> select count(*) from tkaqqtpeqt;

COUNT(*)

0

1 row selected.

SVRMGR> select count(*) from aq\$_tkaqqtpeqt_i;

COUNT(*)

0

1 row selected.

SVRMGR>

SVRMGR> select count(*) from tkaqqtprl;

COUNT(*)

0

1 row selected.

SVRMGR> select count(*) from aq\$_tkaqqtprl_i;

COUNT(*)

0

1 row selected.

SVRMGR>

SVRMGR> Rem create two queues in each queue table


```

SVRMGR>
SVRMGR> Rem create two queues in tkaqqtdef
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q1def', queue_table =>
'sys.tkaqqtdef', max_retries => 2, comment => 'queue 1 in tkaqqtdef');
Statement processed.
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q2def', queue_table =>
'sys.tkaqqtdef', max_retries => 2, comment => 'queue 2 in tkaqqtdef');
Statement processed.
SVRMGR>
SVRMGR> Rem create two queues in tkaqqtpeqt
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q1peqt', queue_table =>
'sys.tkaqqtpeqt', max_retries => 2, comment => 'queue 1 in tkaqqtpeqt');
Statement processed.
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q2peqt', queue_table =>
'sys.tkaqqtpeqt', max_retries => 2, comment => 'queue 2 in tkaqqtpeqt');
Statement processed.
SVRMGR>
SVRMGR> Rem create two queues in tkaqqtprl
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q1prl', queue_table =>
'sys.tkaqqtprl', max_retries => 2, comment => 'queue 1 in tkaqqtprl');
Statement processed.
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'q2prl', queue_table =>
'sys.tkaqqtprl', max_retries => 2, comment => 'queue 2 in tkaqqtprl');
Statement processed.
SVRMGR>
SVRMGR> Rem create an exception queue in the tkaqqtdef table
SVRMGR>
SVRMGR> execute dbms_aqadm.create_queue(queue_name => 'exceptionq', queue_table =>
in tkaqqtdef');
Statement processed.
SVRMGR>
SVRMGR> Rem Create procedure to check the list of subscribers for each queue
SVRMGR>
SVRMGR> CREATE OR REPLACE PROCEDURE TKAQ_SUBSCRIBERS(qname VARCHAR2) AS
2>
3> subs      dbms_aqadm.aq$_subscriber_list_t;
4> nsubs     BINARY_INTEGER;
5> i         BINARY_INTEGER;
6>
7> begin
8>
9>   subs := dbms_aqadm.queue_subscribers(qname);
10>
11>   dbms_output.put_line(qname);
12>   dbms_output.put_line('-----');
13>
14>   nsubs := subs.COUNT;
15>   FOR i IN 0..nsubs-1 LOOP
16>     IF subs(i) IS NOT NULL THEN
17>       dbms_output.put_line('--> ' || subs(i).name);
18>     END IF;
19>   END LOOP;
20>
21> end TKAQ_SUBSCRIBERS;
22> /
Statement processed.

```

```

SVRMGR>
SVRMGR> CREATE OR REPLACE PROCEDURE TKAQ_HISTORY(enqmsgsid RAW) AS
2>
3> hist          sys.aq$_history;
4> nsubs         BINARY_INTEGER;
5> i             BINARY_INTEGER;
6>
7> begin
8>
9>   select history into hist
10>  from   tkaqqtdef
11>  where  msgsid = enqmsgsid;
12>
13>
14>   dbms_output.put_line('-----');
15>   IF hist IS NOT NULL THEN
16>     nsubs := hist.COUNT;
17>     dbms_output.put_line(nsubs);
18>     FOR i IN 1..nsubs LOOP
19>       IF hist(i) IS NOT NULL THEN
20>         dbms_output.put_line('APP --> ' || hist(i).consumer);
21>         IF hist(i).transaction_id IS NOT NULL THEN
22>           dbms_output.put_line('TXN --> ' || hist(i).transaction_id);
23>           dbms_output.put_line('DEQ_USER --> ' || hist(i).deq_user);
24>         ELSE
25>           dbms_output.put_line('TXN --> ');
26>         END IF;
27>       END IF;
28>     END LOOP;
29>   END IF;
30>   dbms_output.put_line('-----');
31>
32> end TKAQ_HISTORY;
33> /

```

MGR-00072: Warning: PROCEDURE TKAQ_HISTORY created with compilation errors.

```

SVRMGR>
SVRMGR>
SVRMGR> Rem check subscribers for queues created
SVRMGR>

```

```

SVRMGR> set serveroutput on
Server Output          ON

```

```

SVRMGR> execute tkaq_subscribers('Q1DEF');
Statement processed.
Q1DEF

```

```

-----
SVRMGR> execute tkaq_subscribers('Q2DEF');
Statement processed.
Q2DEF

```

```

-----
SVRMGR> execute tkaq_subscribers('Q1PEQT');
Statement processed.
Q1PEQT

```

```

-----
SVRMGR> execute tkaq_subscribers('Q2PEQT');
Statement processed.
Q2PEQT

```

```

-----
SVRMGR> execute tkaq_subscribers('Q1PRI');
Statement processed.
Q1PRI

```

```

-----
SVRMGR> execute tkaq_subscribers('Q2PRI');

```

Statement processed.
Q2PRI

```
-----
SVRMGR>
SVRMGR> Rem add some default subscribers for each queue.
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app1_qldef', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
6> end;
7> /
```

Statement processed.

```
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_qldef', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
6> end;
7> /
```

Statement processed.

```
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app3_qldef', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
6> end;
7> /
```

Statement processed.

```
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app4_qldef', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
6> end;
7> /
```

Statement processed.

```
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app5_qldef', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
6> end;
7> /
```

Statement processed.

```
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app6_qldef', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.qldef',appl_qldef);
6> end;
7> /
```

Statement processed.

```
SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
```

```

3> begin
4> appl_qldef := sys.aq$_agent('app1_q2def', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2def',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_q2def', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2def',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app1_q1peqt', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q1peqt',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_q1peqt', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q1peqt',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app1_q2peqt', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2peqt',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_q2peqt', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2peqt',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app1_q1pri', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q1pri',appl_qldef);
6> end;
7> /
statement processed.

```

```

SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_qlpri', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.qlpri',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app1_q2pri', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2pri',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR> declare
2> appl_qldef sys.aq$_agent;
3> begin
4> appl_qldef := sys.aq$_agent('app2_q2pri', NULL, NULL);
5> dbms_aqadm.add_subscriber('sys.q2pri',appl_qldef);
6> end;
7> /
Statement processed.
SVRMGR>
SVRMGR> Rem check subscribers for queues created
SVRMGR>
SVRMGR> execute tkaq_subscribers('Q1DEF');
Statement processed.
Q1DEF
-----
--> APP1_Q1DEF
--> APP2_Q1DEF
--> APP3_Q1DEF
--> APP4_Q1DEF
--> APP5_Q1DEF
--> APP6_Q1DEF
SVRMGR> execute tkaq_subscribers('Q2DEF');
Statement processed.
Q2DEF
-----
--> APP1_Q2DEF
--> APP2_Q2DEF
SVRMGR> execute tkaq_subscribers('Q1PEQT');
Statement processed.
Q1PEQT
-----
--> APP1_Q1PEQT
--> APP2_Q1PEQT
SVRMGR> execute tkaq_subscribers('Q2PEQT');
Statement processed.
Q2PEQT
-----
--> APP1_Q2PEQT
--> APP2_Q2PEQT
SVRMGR> execute tkaq_subscribers('Q1PRI');
Statement processed.
Q1PRI

```

--> APP1_Q1PRI
--> APP2_Q1PRI
SVRMGR> execute tkaq_subscribers('Q2PRI');
Statement processed.
Q2PRI

--> APP1_Q2PRI
--> APP2_Q2PRI
SVRMGR>
SVRMGR> Rem start the queues
SVRMGR> execute dbms_aqadm.start_queue(queue_name => 'sys.q1def');
Statement processed.
SVRMGR> execute dbms_aqadm.start_queue(queue_name => 'sys.q2def');
Statement processed.
SVRMGR> execute dbms_aqadm.start_queue(queue_name => 'sys.q1peqt');
Statement processed.
SVRMGR> execute dbms_aqadm.start_queue(queue_name => 'sys.q2peqt');
Statement processed.
SVRMGR> execute dbms_aqadm.start_queue(queue_name => 'sys.q1pri');
Statement processed.
SVRMGR> execute dbms_aqadm.start_queue(queue_name => 'sys.q2pri');
Statement processed.
SVRMGR>

```

Echo
SVRMGR> connect tkaquser/tkaquser
Connected.
SVRMGR> set serveroutput on
Server Output
SVRMGR>
SVRMGR> create or replace procedure tkaq_navenq(priority in number) as
2>   enq_userdata sys.message;
3>   enq_msgid    raw(16);
4>   enqopt       dbms_aq.enqueue_options_t;
5>   msgprop      dbms_aq.message_properties_t;
6>
7>   begin
8>
9>   enq_userdata := sys.message(priority, 'HELLO, WORLD!');
10>  msgprop.priority := priority;
11>  dbms_aq.enqueue('sys.q2pri', enqopt, msgprop, enq_userdata, enq_msgid);
12> end;
13> /
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navenq(1);
Statement processed.
SVRMGR> execute tkaq_navenq(2);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navenq(3);
Statement processed.
SVRMGR> execute tkaq_navenq(4);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navenq(5);
Statement processed.
SVRMGR> execute tkaq_navenq(6);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navenq(7);
Statement processed.
SVRMGR> execute tkaq_navenq(8);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navenq(9);
Statement processed.
SVRMGR> execute tkaq_navenq(10);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navenq(11);
Statement processed.
SVRMGR> execute tkaq_navenq(12);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>

```

```

SVRMGR> execute tkaq_navenq(13);
Statement processed.
SVRMGR> execute tkaq_navenq(14);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_navenq(15);
Statement processed.
SVRMGR> execute tkaq_navenq(16);
Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>
SVRMGR> create or replace procedure tkaq_navdeq(consumer IN VARCHAR2) as
2> deq_userdata sys.message;
3> deq_msgid raw(16);
4> deqopt dbms_aq.dequeue_options_t;
5> msgprop dbms_aq.message_properties_t;
6>
7> begin
8>
9> deqopt.wait := DBMS_AQ.NO_WAIT;
10> deqopt.consumer_name := consumer;
11> deqopt.navigation := DBMS_AQ.FIRST_MESSAGE;
12> deqopt.dequeue_mode := DBMS_AQ.BROWSE;
13>
14> FOR i in 1..9 loop
15>   dbms_aq.dequeue('sys.q2pri', deqopt, msgprop, deq_userdata, deq_msgid);
16>   dbms_output.put_line('Message: ' || deq_userdata.id
17>                        || ':' || deq_userdata.data);
18>   commit;
19>   deqopt.navigation := DBMS_AQ.NEXT_MESSAGE;
20> END LOOP;
21>
22> end;
23> /

```

Statement processed.

```

SVRMGR>
SVRMGR> execute tkaq_navdeq('app1_q2pri');
Statement processed.

```

```

Message: 1:HELLO, WORLD!
Message: 2:HELLO, WORLD!
Message: 3:HELLO, WORLD!
Message: 4:HELLO, WORLD!
Message: 5:HELLO, WORLD!
Message: 6:HELLO, WORLD!
Message: 7:HELLO, WORLD!
Message: 8:HELLO, WORLD!
Message: 9:HELLO, WORLD!

```

```

SVRMGR> execute tkaq_navdeq('app2_q2pri');
Statement processed.

```

```

Message: 1:HELLO, WORLD!
Message: 2:HELLO, WORLD!
Message: 3:HELLO, WORLD!
Message: 4:HELLO, WORLD!
Message: 5:HELLO, WORLD!
Message: 6:HELLO, WORLD!
Message: 7:HELLO, WORLD!
Message: 8:HELLO, WORLD!
Message: 9:HELLO, WORLD!

```

```

SVRMGR>
SVRMGR> create or replace procedure tkaq_navdeq(consumer IN VARCHAR2) as

```



```

2> deq_userdata sys.message;
3> deq_msgid    raw(16);
4> deqopt       dbms_aq.dequeue_options_t;
5> msgprop      dbms_aq.message_properties_t;
6>
7> begin
8>
9> deqopt.wait := DBMS_AQ.NO_WAIT;
10> deqopt.consumer_name := consumer;
11> deqopt.navigation := DBMS_AQ.FIRST_MESSAGE;
12>
13>   FOR i in 1..19 loop
14>       dbms_aq.dequeue('sys.q2pri', deqopt, msgprop, deq_userdata, deq_msgid);
15>       dbms_output.put_line('Message: ' || deq_userdata.id
16>                             || ':' || deq_userdata.data);
17>       commit;
18>       deqopt.navigation := DBMS_AQ.NEXT_MESSAGE;
19>   END LOOP;
20>
21> end;
22> /
statement processed.
SVRMGR>
SVRMGR> execute tkaq_navdeq('app1_q2pri');
ORA-25228: timeout in dequeue from SYS.Q2PRI while waiting for a message
ORA-06512: at "SYS.DBMS_AQ", line .*
ORA-06512: at "TKAQUSER.TKAQ_NAVDEQ", line .*
ORA-06512: at line .*
Message: 1:HELLO, WORLD!
Message: 2:HELLO, WORLD!
Message: 3:HELLO, WORLD!
Message: 4:HELLO, WORLD!
Message: 5:HELLO, WORLD!
Message: 6:HELLO, WORLD!
Message: 7:HELLO, WORLD!
Message: 8:HELLO, WORLD!
Message: 9:HELLO, WORLD!
Message: 10:HELLO, WORLD!
Message: 11:HELLO, WORLD!
Message: 12:HELLO, WORLD!
Message: 13:HELLO, WORLD!
Message: 14:HELLO, WORLD!
Message: 15:HELLO, WORLD!
Message: 16:HELLO, WORLD!
SVRMGR> execute tkaq_navdeq('app2_q2pri');
ORA-25228: timeout in dequeue from SYS.Q2PRI while waiting for a message
ORA-06512: at "SYS.DBMS_AQ", line .*
ORA-06512: at "TKAQUSER.TKAQ_NAVDEQ", line .*
ORA-06512: at line .*
Message: 1:HELLO, WORLD!
Message: 2:HELLO, WORLD!
Message: 3:HELLO, WORLD!
Message: 4:HELLO, WORLD!
Message: 5:HELLO, WORLD!
Message: 6:HELLO, WORLD!
Message: 7:HELLO, WORLD!
Message: 8:HELLO, WORLD!
Message: 9:HELLO, WORLD!
Message: 10:HELLO, WORLD!
Message: 11:HELLO, WORLD!
Message: 12:HELLO, WORLD!
Message: 13:HELLO, WORLD!
Message: 14:HELLO, WORLD!

```

Message: 15:HELLO, WORLD!
Message: 16:HELLO, WORLD!
SVRMGR>

```

Echo
SVRMGR> connect sys/knl_test7 as sysdba
Connected.
SVRMGR> set serveroutput on
Server Output
SVRMGR>
SVRMGR>
SVRMGR>
SVRMGR>
SVRMGR> CREATE OR REPLACE PROCEDURE TKAQ_MBASICENQ(id IN NUMBER, text IN VARCHAR2)
AS

```

```

2> msgprop          dbms_aq.message_properties_t;
3> enqopt            dbms_aq.enqueue_options_t;
4> enq_msgid         raw(16);
5> enq_userdata      message;
6>
7> begin
8>
9> enq_userdata := message(id, text);
10> dbms_aq.enqueue(
11>     queue_name => 'sys.q2def',
12>     enqueue_options => enqopt,
13>     message_properties => msgprop,
14>     payload => enq_userdata,
15>     msgid => enq_msgid);
16>
17> end;
18> /

```

Statement processed.

```

SVRMGR>
SVRMGR> CREATE OR REPLACE PROCEDURE TKAQ_MBASICDEQ(subscriber IN VARCHAR2) AS

```

```

2>
3> dequeue_options   dbms_aq.dequeue_options_t;
4> message_properties dbms_aq.message_properties_t;
5> deq_userdata      sys.message;
6> deq_msgid         raw(16);
7> begin
8> dequeue_options.consumer_name := subscriber;
9> dequeue_options.navigation := DBMS_AQ.FIRST_MESSAGE;
10> dequeue_options.wait := 1;
11> dbms_aq.dequeue(queue_name=> 'sys.q2def',
12>     dequeue_options=>dequeue_options,
13>     message_properties=>message_properties,
14>     payload=>deq_userdata,
15>     msgid=>deq_msgid);
16> commit;
17>
18> dbms_output.put_line('MSG-> ' || deq_userdata.id || ' ' ||
deq_userdata.data);
19>
20> end;
21> /

```

Statement processed.


```

SVRMGR>
SVRMGR> Rem enqueue twelve messages each subscriber should get two.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(1, 'First Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(2, 'Second Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(3, 'Third Message');

```

Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(4, 'Fourth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(5, 'Fifth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(6, 'Sixth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(7, 'Seventh Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(8, 'Eight Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(9, 'Ninth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(10, 'Tenth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(11, 'Eleventh Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(12, 'Twelveth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(13, 'Thirteenth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(14, 'Fourteenth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(15, 'Fifteenth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(16, 'Sixteenth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(17, 'Seventeenth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(18, 'Eighteenth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(19, 'Nineteenth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(20, 'Twentyth Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(21, 'Twentyfirst Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(22, 'Twenty2nd Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(23, 'Twenty3rd Message');
Statement processed.
SVRMGR>
SVRMGR> execute tkaq_mbasicenq(24, 'Twenty4th Message');

Statement processed.
SVRMGR> commit;
Statement processed.
SVRMGR>



Echo
SVRMGR> connect sys/knl_test7 as sysdba
Connected.
SVRMGR> set serveroutput on
Server Output ON
SVRMGR>
SVRMGR> Rem now do dequeues
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 1 First Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 1 First Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 2 Second Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 2 Second Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 3 Third Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 3 Third Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 4 Fourth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 4 Fourth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 5 Fifth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 5 Fifth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 6 Sixth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 6 Sixth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 7 Seventh Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 7 Seventh Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 8 Eight Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 8 Eight Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 9 Ninth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 9 Ninth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.

```


MSG-> 10 Tenth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 10 Tenth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 11 Eleventh Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 11 Eleventh Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 12 Twelveth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 12 Twelveth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 13 Thirteenth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 13 Thirteenth Message
SVRMGR> commit;
Statement processed.
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 14 Fourteenth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 15 Fifteenth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 16 Sixteenth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 17 Seventeenth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 18 Eighteenth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 19 Nineteenth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 20 Twentieth Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 21 Twentyfirst Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 22 Twenty2nd Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 23 Twenty3rd Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
Statement processed.
MSG-> 24 Twenty4th Message
SVRMGR> execute tkaq_mbasicdeq('app1_q2def');
ORA-25228: timeout in dequeue from SYS.Q2DEF while waiting for a message
ORA-06512: at "SYS.DBMS_AQ", line .*
ORA-06512: at "SYS.TKAQ_MBASICDEQ", line .*
ORA-06512: at line .*
SVRMGR> commit;

```

Statement processed.
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 14 Fourteenth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 15 Fifteenth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 16 Sixteenth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 17 Seventeenth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 18 Eighteenth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 19 Nineteenth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 20 Twentieth Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 21 Twentyfirst Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 22 Twenty2nd Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 23 Twenty3rd Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
Statement processed.
MSG-> 24 Twenty4th Message
SVRMGR> execute tkaq_mbasicdeq('app2_q2def');
ORA-25228: timeout in dequeue from SYS.Q2DEF while waiting for a message
ORA-06512: at "SYS.DBMS_AQ", line .*
ORA-06512: at "SYS.TKAQ_MBASICDEQ", line .*
ORA-06512: at line .*
SVRMGR> commit;
Statement processed.

[REDACTED]
ON

```
Echo
SVRMGR>
SVRMGR> connect sys/knl_test7 as sysdba
Connected.
SVRMGR>
SVRMGR> Rem cleanup the queue tables
SVRMGR>
SVRMGR> execute dbms_aqadm.drop_queue_table('sys.tkaqqtdef', TRUE);
Statement processed.
SVRMGR> execute dbms_aqadm.drop_queue_table('sys.tkaqqtpeqt', TRUE);
Statement processed.
SVRMGR> execute dbms_aqadm.drop_queue_table('sys.tkaqqtprpri', TRUE);
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> Rem connect as sys and drop the types
SVRMGR> connect sys/knl_test7 as sysdba
Connected.
SVRMGR>
SVRMGR> drop type message;
Statement processed.
SVRMGR>
SVRMGR>
SVRMGR> Rem disable aq logins
SVRMGR>
SVRMGR> drop user tkaquser cascade;
Statement processed.
SVRMGR> drop user tkaqadmn cascade;
Statement processed.
SVRMGR>
SVRMGR>
```



```
execute tkaq_mbasicdeq('app1_q2def');
execute tkaq_mbasicdeq('app1_q2def');
commit;
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
execute tkaq_mbasicdeq('app2_q2def');
commit;
```

EXHIBIT F

Short regress has 0 dif - run with use_ism=false.

tk0irddl.suc	tkagrawt.suc	tkoolqyi.suc	tkprddls.suc	tkqrords.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdrpls1.suc	tkoqbrqr.suc	tkprdepl.suc	tkqrsels.suc
tkaqbdq2.suc	tkgrsecs.suc	tkp83cl.suc	tkprdep2.suc	tkqrtres.suc
tkaqbeql.suc	tkhodbck.suc	tkpasr81.suc	tkprdep3.suc	tkqrwhe.suc
tkaqbeq2.suc	tkhomrg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxeisr.suc
tkaqdqbl.suc	tkhoucmr.suc	tkpbd1.suc	tkprdep5.suc	tkrcrash.suc
tkaqdqbl2.suc	tkigfsrt.suc	tkpmsrl.suc	tkprdep6.suc	tkrcrsha.suc
tkaqdqba.suc	tkirenbs.suc	tkpmsrnl.suc	tkprexec.suc	tkrmlite.suc
tkaqdqrc.suc	tkirinds.suc	tkpopro.suc	tkprmisc.suc	tkrmliti.suc
tkaqmbdq.suc	tkoodcat.suc	tkpotta.suc	tkprsq1p.suc	tkrsrsbs.suc
tkaqmbei.suc	tkoodcta.suc	tkpgcafs.suc	tkprtime.suc	tktrclus.suc
tkaqmnav.suc	tkoodctn.suc	tkpgdups.suc	tkprtimm.suc	tktrmmps.suc
tkaqnsbl1.suc	tkoodtcl.suc	tkpggrps.suc	tkpuori.suc	tktrnaos.suc
tkaqnsb2.suc	tkoodtci.suc	tkpqjois.suc	tkqrcats.suc	tktrtabs.suc
tkaqnsb3.suc	tkoodtcm.suc	tkpqmain.suc	tkqrdats.suc	tktrvies.suc
tkaqnsb4.suc	tkoodtir.suc	tkpgords.suc	tkqrexps.suc	tkxabrch.suc
tkaqnsb5.suc	tkoodxu0.suc	tkpqgrgs.suc	tkqrgtps.suc	tkzrls00.suc
tkaqoci2.suc	tkoolqyc.suc	tkpqwhrs.suc	tkqrjois.suc	
tkaqrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuls.suc	

Long regress will not be started as we need machines for other tests.

Transaction: ntang_bug-733938

Transaction: ykunitom_bug-704908_1

kpodp.c@@/main/3
 Bug 704908
 kdbl.c@@/main/115
 Bug 704908
 kdbl.h@@/main/26
 kta.c@@/main/3
 Bug 704908
 klc.c@@/main/102
 klc11.c@@/main/16
 opiul.c@@/main/35

Transaction: pong_bug-713950_1

ksdx0.h@@/main/4
 [REDACTED]

ksdx.c@@/main/22
 [REDACTED]

Transaction: weiwang_bug-718209

kkdl.c@@/main/378
 [REDACTED]

[REDACTED] Merge From /vobs/rdbms/src/server/dict/dictlkup/kkdl.c@@/main/377

kkfi.c@@/main/18
 [REDACTED]

[REDACTED] Merge From /vobs/rdbms/src/server/optim/cbo/kkfi.c@@/main/17

kda.c@@/main/117
 [REDACTED]

kau.c@@/main/97
 [REDACTED]

qerit.c@@/main/41
 [REDACTED]

qkdrv.c@@/main/224
 [REDACTED]

[REDACTED] Merge From /vobs/rdbms/src/server/sqlxexec/rwsalloc/qkdrv.c@@/main/223

klcbs.h@@/main/47
 [REDACTED]

[REDACTED] check for disabled flag

klc.c@@/main/101
 [REDACTED]

Transaction: ato_make_user_queue_table_views

catqueue.sql@@/main/34
 [REDACTED]

[REDACTED] create all_queue_tables views

Makefile@@/main/9
 [REDACTED]

[REDACTED] add prvtagin.sql

prvtagin.sql@@/main/3
 [REDACTED]

[REDACTED] add java interface

Transaction: smuralid_bug-718348

ttccap.c@@/main/4
 [REDACTED]

[REDACTED] ttccapSnd: don't send if null caps

koka.c@@/main/38
 [REDACTED]

[REDACTED] Merge From /vobs/rdbms/src/server/objsupp/objdata/koka.c@@/main/37

kpo.h@@/main/21
 [REDACTED]

[REDACTED] make kpoRTcaps an SGA variable

opidrv.c@@/main/81
 [REDACTED]

[REDACTED] opidrv, opiscb: set hstmrtcap to null if SGA's not initialized

opiino.c@@/main/35
 [REDACTED]

[REDACTED] opiino: set hstmrtcap to kpoRTcaps after mapping in SGA

opirip.c@@/main/30
 [REDACTED]

[REDACTED] Merge From /vobs/rdbms/src/server/progint/opi/opirip.c@@/main/29

opitsk.c@@/main/67
 [REDACTED]

[REDACTED] Merge From /vobs/rdbms/src/server/progint/opi/opitsk.c@@/main/66

Transaction: rshaikh_fix_downgrade

c0801030.sql@@/main/10
 [REDACTED]

[REDACTED] drop type rowset

d0800050.sql@@/main/34
 [REDACTED]

[REDACTED] dont drop ustats
d0801030.sql@@/main/7
[REDACTED]: fix

Transaction: najain_658136-1

uacdef.h@@/main/20
[REDACTED] remove UACFNLMX
kxs.c@@/main/151
[REDACTED] remove UACFNLBV
kks.c@@/main/316
[REDACTED] remove UACFNLBV
opix.c@@/main/82
[REDACTED] remove UACFNLBV

Transaction: thchang_templob_rdwr

kokl.c@@/main/64
[REDACTED] Merge From /vobs/rdbms/src/server/objsupp/objdata/kokl.c@@/main/63
kokl.c@@/main/21
[REDACTED] templob

Transaction: ramkrish_bug-685852_2

opitsk.c@@/main/68
[REDACTED] opitsk: bug 685852 - BFILE check

Transaction: lkaplan_row_cln_err

AllA.java@@/main/9
[REDACTED]
RefGrp.java@@/main/5
[REDACTED]
RepAPI.java@@/main/9
[REDACTED]

Transaction: sichandr_bug-729312

kkt.c@@/main/108
[REDACTED] #729312 : fix error message for DATABASE triggers

Transaction: awitkows_bug-732881

vop.c@@/main/135
[REDACTED] complex view & correlated var

Transaction: najain_bug-730182

kql.c@@/main/240
[REDACTED] fix 730182

Transaction: heneman_lrg27341

[REDACTED]

kmc.c@@/main/100

LRG 27341: correct order of interruptable test

Transaction: ntang_bug-737564

k2g.c@@/main/52

b

Out of Transaction changes:

.depprod@@/main/solaris/227

: update plsqli, precomp, sqlplus, ordts, ldap

.laperlog@@/main/264

: new log

.manifest@@/main/solaris/271

: new label

msg@@/main/5

Add new qsmXX.msg translations for 8.1.4

Added file element "qsmd.msg".

Added file element "qsmf.msg".

Added file element "qsmja.msg".

qsmd.msg@@/main/0

qsmf.msg@@/main/0

qsmja.msg@@/main/0

joint@@/main/6

Add 8.1.4 RepAPIError.properties translation entries

Added file element "RepAPIErrorD.properties".

Added file element "RepAPIErrorF.properties".

Added file element "RepAPIErrorJA.properties".

RepAPIErrorD.properties@@/main/0

RepAPIErrorF.properties@@/main/0

RepAPIErrorJA.properties@@/main/0

=====

Short regress has 0 dif - run with use_ism=false.

tk0irddl.suc	tkagrawt.suc	tkoolqyi.suc	tkprddls.suc	tkqrords.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdplsl.suc	tkogbrqr.suc	tkprdepl.suc	tkqrsels.suc
tkaqbdq2.suc	tkgrsecs.suc	tkp83cl.suc	tkprdep2.suc	tkqrtres.suc
tkaqbeql.suc	tkhodbck.suc	tkpasr8l.suc	tkprdep3.suc	tkqrwhes.suc
tkaqbeq2.suc	tkhomrg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxeisz.suc
tkaqdqb1.suc	tkhoucmr.suc	tkpbd1.suc	tkprdep5.suc	tkrcrash.suc
tkaqdqb2.suc	tkigfsrt.suc	tkpmsrl.suc	tkprdep6.suc	tkrcrsha.suc
tkaqdqb3.suc	tkirenbs.suc	tkpmsrnl.suc	tkprexec.suc	tkrmlite.suc
tkaqdqr.c.suc	tkirinds.suc	tkpopro.suc	tkprmisc.suc	tkrmliti.suc
tkaqmbdq.suc	tkoodcat.suc	tkpotta.suc	tkprsq1p.suc	tkrsrsubs.suc
tkaqmbei.suc	tkoodcta.suc	tkpgcafs.suc	tkprttime.suc	tktrclus.suc
tkaqmnav.suc	tkoodctn.suc	tkpgdups.suc	tkprtimm.suc	tktrmmps.suc
tkaqnsb1.suc	tkoodtcl.suc	tkpggrps.suc	tkpuori.suc	tktrnaos.suc
tkaqnsb2.suc	tkoodtci.suc	tkpgjois.suc	tkqrcats.suc	tktrtabs.suc
tkaqnsb3.suc	tkoodtcm.suc	tkpqmain.suc	tkqrdats.suc	tktrvies.suc
tkaqnsb4.suc	tkoodtir.suc	tkpgords.suc	tkqrexps.suc	tkxabrch.suc
tkaqnsb5.suc	tkoodxu0.suc	tkpqrgrs.suc	tkqrgrrs.suc	tkzrlso0.suc
tkaqoci2.suc	tkoolqyc.suc	tkpqwhrs.suc	tkqrjois.suc	
tkagrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuis.suc	

Long regress has been started.

Transaction: rjenkins_bug-702786

```

kkdc.c@@/main/85
Merge From /vobs/rdbms/src/server/dict/dictlkup/kkdc.c@@/main/84
kpc.c@@/main/17
more stuff
kcpam.h@@/main/13
702786: make kcpamKRange return fragment numbers
kdic.c@@/main/131
Merge From /vobs/rdbms/src/server/ram/index/kdic.c@@/main/130
kcpam.c@@/main/19
702786: merge changes from andre
kcpam0.h@@/main/8
702786: merging in andre's changes
delexe.c@@/main/163
try again
kxcc.h@@/main/15
Merge From /vobs/rdbms/src/server/sqlang/if/kxcc.h@@/main/14
kxcc.c@@/main/23
Merge From /vobs/rdbms/src/server/sqlang/integ/kxcc.c@@/main/22
xty.c@@/main/189
Merge From /vobs/rdbms/src/server/sqlang/typeconv/xty.c@@/main/188

```

Transaction: thoang_partobj_bugs

```

atb.c@@/main/250
koke.h@@/main/28
Add fragno argument to kokeicd2m()

```



```
- kkbl.c@@/main/38
[REDACTED] kkblclsi: setup linfo for varray stored as lob
koke.c@@/main/45
[REDACTED]: Pass fragno to kokeicd2m()
kaf.c@@/main/78
[REDACTED] Pass fragno to kokeicd2m()
updexe.c@@/main/231
[REDACTED] Pass fragno to kokeicd2m
```

Out of Transaction changes:

```
.labellog@@/main/261
[REDACTED] new log
.manifest@@/main/solaris/268
[REDACTED] new label
.labellog@@/main/260
[REDACTED] Correcting root directory entries for 980925 label.
```

=====

This label was compiled with -xprofile=use option.

Short regress has 0 dif - run with use_ism=false.

tk0irddl.suc	tkagrawt.suc	tkoolqyi.suc	tkprddl.s.suc	tkqrords.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdrplsl.suc	tkogbrqr.suc	tkprdepl.suc	tkqrsels.suc
tkaqbdq2.suc	tkgrsecs.suc	tkp83cl.suc	tkprdep2.suc	tkqrtres.suc
tkaqbeql.suc	tkhodbck.suc	tkpasr8l.suc	tkprdep3.suc	tkqrwhes.suc
tkaqbeq2.suc	tkhomrg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxeisr.suc
tkaqddb1.suc	tkhoucmr.suc	tkpbd1.suc	tkprdep5.suc	tkrcrash.suc
tkaqddb2.suc	tkigfsrt.suc	tkpmsrl.suc	tkprdep6.suc	tkrcrsha.suc
tkaqdgb2.suc	tkirenbs.suc	tkpmsrnl.suc	tkprexec.suc	tkrmlite.suc
tkaqdgrc.suc	tkirinds.suc	tkpopro.suc	tkprmisc.suc	tkrmliti.suc
tkaqmbdq.suc	tkoodcat.suc	tkpotta.suc	tkprsqpl.suc	tkrsrsbs.suc
tkaqmbei.suc	tkoodcta.suc	tkpgcafs.suc	tkprtime.suc	tktrclus.suc
tkaqmnav.suc	tkoodctn.suc	tkpgdups.suc	tkprtimm.suc	tktrmnp.suc
tkaqnsb1.suc	tkoodtcl.suc	tkpggrps.suc	tkpuori.suc	tktrnaos.suc
tkaqnsb2.suc	tkoodtci.suc	tkpqjois.suc	tkqrcats.suc	tktrtabs.suc
tkaqnsb3.suc	tkoodtcm.suc	tkpqmain.suc	tkqrdats.suc	tktrvies.suc
tkaqnsb4.suc	tkoodtir.suc	tkpgords.suc	tkqrexps.suc	tkxabrch.suc
tkaqnsb5.suc	tkoodxu0.suc	tkpqrgrs.suc	tkqrgrrs.suc	tkxafini.suc
tkaqoci2.suc	tkoolqyc.suc	tkpqwhrs.suc	tkqrjois.suc	tkxainit.suc
tkaqrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuls.suc	tkzrlso0.suc

Long regress will be started tomorrow evening.

Transaction: ramkrish_bugfix-varray

kkbl.c@@/main/30

kkblcisi: fix varray bug introduced as part of atbmov

Transaction: snutakki_bug-511898_5

gerix.c@@/main/128

Fix regression from merge of 511898

Transaction: syeung_bug-702172_1

kda.c@@/main/113

Transaction: mkrishna_fix_objview_bugs

```

kxti.h@@/main/5
[REDACTED] add KXTDF_NTT_TRIGGER
kxti.c@@/main/14
[REDACTED]
kxto.c@@/main/22
[REDACTED]: fix bug 704081
kkdo.h@@/main/20
[REDACTED] make lint happy
kokv.c@@/main/20
[REDACTED] fix core dumps when invalid type is given
nsqbc.c@@/main/53
[REDACTED] nsomrg: set up froljc for dependent tables
xty.c@@/main/180
[REDACTED]
cvw.c@@/main/105
[REDACTED] fix cvwosfor
delexe.c@@/main/155
[REDACTED] change kxtifrw calling

```

Transaction: amozes_lrg-25369

```

dbsdrv.c@@/main/162
[REDACTED] check if database is open before calling ksxshut

```

Transaction: sdas_bug-703264

```

kau.c@@/main/90
[REDACTED] partitioned iot: fix bug 703264
kdu.c@@/main/115
[REDACTED] kdudcp(): replace kduuskflag with kduusdflag
kau.h@@/main/36
[REDACTED] name/interface chg: kauboikey->kauibokey, kaubnikey->kauibnkey
delexe.c@@/main/156
* [REDACTED] iot: set kduikauc in delini()
updexe.c@@/main/221
[REDACTED] iot: remove kdkreb call, kauibokey did its job
qerfu.c@@/main/23
[REDACTED] iot: set kduikauc in qerfuInitFrame()

```

Transaction: bnnguyen_bug-678044

```

psdicd.c@@/main/174
[REDACTED] bug678044

```

Transaction: svedala_create_part_demo

```

demo@@/main/19
[REDACTED] Added file element "cdemopl.b.c".
[REDACTED] Added file element "cdemopl.b.sql".
[REDACTED] Added file element "cdemopl.b.h".
[REDACTED] Added file element "cdemopl.b.dat".
[REDACTED] Removed file element "cdemopl.b.h".
[REDACTED] Removed file element "cdemopl.b.sql".
[REDACTED] Removed file element "cdemopl.b.dat".
cdemopl.b.c@@/main/1
[REDACTED]

```

Transaction: amganeshtprundo-noret

ktpr.c@@/main/13

[REDACTED]: ktprundo doesnt return

Transaction: thoang_merge_696471

kgl.h@@/main/205

[REDACTED]: Removed kglcldp

kgl2.c@@/main/109

[REDACTED]: Removed kglcldp

kkm.c@@/main/380

[REDACTED]:

kkpod.c@@/main/27

[REDACTED]:

atb.c@@/main/228

[REDACTED]:

cvw.c@@/main/106

[REDACTED]:

dix.c@@/main/52

[REDACTED]:

qkadv.c@@/main/209

[REDACTED]:

Transaction: nmacnaug_merge_9

kcbz.h@@/main/46

[REDACTED]: do not use kcbhbha field directly

kcl.c@@/main/113

[REDACTED]: rename structure element to avoid conflict

kcl0.h@@/main/25

[REDACTED]: rename structure element to avoid conflict

Transaction: st_plsql_smkrishn_no_rebind

kkxwtp.c@@/main/192

[REDACTED]: Merge From /vobs/rdbms/src/server/progint/opi/kkxwtp.c@@/main/188

Transaction: nmacnaug_fix_24

kcl.h@@/main/53

[REDACTED]: add exclusive hint

kcbz.c@@/main/119

[REDACTED]: use exclusive hint

kcl2.h@@/main/34

[REDACTED]: send class correctly to bsp

kcl.c@@/main/111

[REDACTED]: send class correctly to bsp

Transaction: lkaplan_change_drop

FDSAccess.java@@/main/5

[REDACTED]

██████████
 FDSObject.java@@/main/5
 ██████████:

Transaction: nmacnaug_lrg-25129

kcl2.h@@/main/35
 ██████████: remove queued upconvert
 kcl.c@@/main/112
 ██████████: remove queued upconvert
 kcl0.h@@/main/24
 ██████████: remove queued upconvert

Transaction: jklein_bug-708701

kdd.c@@/main/83
 ██████████: bug 708701 - maintain col_list between kudnkc calls.

Transaction: st_plsql_smkrishn_backout_rdbms

kkxwtp.c@@/main/194
 ██████████: Temporarily back out no rebind

Transaction: mcusson_logmnr_nfy

krv.h@@/main/11
 ██████████: Fix short regress problem caused by krvnfy()
 krvr.c@@/main/6
 ██████████: Fix short regress problem caused by krvnfy()

Out of Transaction changes:

.depprod@@/main/solaris/184
 ██████████ update nlstrl, network, spatial, slax, oracore, /vobs/oracle & precomp
 .labellog@@/main/212
 ██████████ new log
 .manifest@@/main/solaris/219
 ██████████: new label
 dbfmig.c@@/main/osds/unix/solaris/1
 ██████████:
 kkxwtp.c@@/main/193
 ██████████ Back out psdscp, psdrcp until pfrrun.c is changed

=====

Short regress has 0 dif - run with use_ism=false.

tk0irddl.suc	tkagrawt.suc	tkoolqyi.suc	tkprddls.suc	tkqrords.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdrpls1.suc	tkogbrqz.suc	tkprdepl.suc	tkqrseis.suc
tkaqbdq2.suc	tkgrsecs.suc	tkp83c1.suc	tkprdep2.suc	tkqrtrres.suc
tkaqbeql.suc	tkhodbck.suc	tkpasr01.suc	tkprdep3.suc	tkqrwhes.suc
tkaqbeq2.suc	tkhomrg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxseisr.suc
tkaqdqb1.suc	tkhoucmr.suc	tkpbd1.suc	tkprdep5.suc	tkrcrash.suc
tkaqdqb2.suc	tkigfsrt.suc	tkpmsr1.suc	tkprdep6.suc	tkrcrsha.suc
tkaqdqba.suc	tkirenbs.suc	tkpmsrnl.suc	tkprexec.suc	tkrmlite.suc
tkaqdqrc.suc	tkirinds.suc	tkpopro.suc	tkprmisc.suc	tkrmliti.suc
tkaqmbdq.suc	tkoodcat.suc	tkpotta.suc	tkprsqpl.suc	tkrsrsubs.suc
tkaqmbei.suc	tkoodcta.suc	tkpgcafs.suc	tkprtime.suc	tktrclul.suc
tkaqmnav.suc	tkoodctn.suc	tkpgdups.suc	tkprtimm.suc	tktrmmps.suc
tkaqnsbl.suc	tkoodtcl.suc	tkpggrps.suc	tkpuori.suc	tktrnaos.suc
tkaqnsb2.suc	tkoodtci.suc	tkpgjois.suc	tkqrcats.suc	tktrtabs.suc
tkaqnsb3.suc	tkoodtcm.suc	tkpgmain.suc	tkqrdats.suc	tktrvies.suc
tkaqnsb4.suc	tkoodtir.suc	tkpgords.suc	tkqrexps.suc	tkxabrch.suc
tkaqnsb5.suc	tkoodxu0.suc	tkpqrgrs.suc	tkqrgrps.suc	tkxafini.suc
tkaqoci2.suc	tkoolqyc.suc	tkpgwhrs.suc	tkqrjois.suc	tkxainit.suc
tkaqrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuls.suc	tkzrls00.suc

Long regress will not be started.

Transaction: smuralid_bugs2

kokq.c@@/main/14

kokqtpo, kokqbpo: propagate OPNOREF

koks.c@@/main/52

koksrcqb: follow qbcnxt only if NOT(is_first)

Transaction: liwong_add_error_23473

e19400.msg@@/main/60

Add 23473

Transaction: dmwong_approlemsg

e24280.msg@@/main/72

add application role err msg

Transaction: bdagevil_ppwj_no_hj_buffout

gerhj.h@@/main/20

change flag BUFFER_QKNJO to NO_BUFFER_QKNJO

qkna.h@@/main/14

change flag BUFFER_QKNJO to NO_BUFFER_QKNJO

gerhj.c@@/main/46

buffer output if BUFFER_QKNJO is set

qkadrv.c@@/main/205

qkadrv2(): add phase to set HJ output buff requirement

qkajoi.c@@/main/133

[REDACTED]: remove every reference to qkanbf()

Transaction: masubram_masubram_null_ref1_1

c0800050.sql@@/main/3
 [REDACTED]
 d0800050.sql@@/main/30
 [REDACTED]
 sql.bsqq@@/main/202
 [REDACTED]
 kkdl.c@@/main/360
 [REDACTED]: Merge From /vobs/rdbms/src/server/dict/dictlkup/kkdl.c@@/main/356
 kkz.h@@/main/46
 [REDACTED]
 kkzd.h@@/main/12
 [REDACTED]:
 kkzf.h@@/main/12
 [REDACTED]
 kkzi.h@@/main/5
 [REDACTED]
 kkzu.h@@/main/10
 [REDACTED]
 kkzv.h@@/main/5
 [REDACTED]
 kkzd.c@@/main/19
 [REDACTED]
 kkzf.c@@/main/26
 [REDACTED]
 kkzi.c@@/main/12
 [REDACTED]
 kkzu.c@@/main/12
 [REDACTED]
 kkzv.c@@/main/15
 [REDACTED]
 prvtsnap.sql@@/main/27
 [REDACTED]
 knt.c@@/main/13
 [REDACTED]: Merge From /vobs/rdbms/src/server/repl/trigger/knt.c@@/main/12
 qsmqutl.c@@/main/5
 [REDACTED]
 kkz.c@@/main/126
 [REDACTED]:
 [REDACTED]

Transaction: bgoyal_bug-692581

e29250.msg@@/main/92
 [REDACTED] add 30555

Transaction: qyu_bug-428835

catalog.sql@@/main/302
 [REDACTED]

Transaction: vkarra_tpcd_changes

kdi.h@@/main/73
 [REDACTED]

```

[REDACTED]: Merge From /vobs/rdbms/src/server/ram/if/kdi.h@@/main/72
kdis.h@@/main/17
[REDACTED]: add variables for leaf and branch splits
kdi.c@@/main/168
[REDACTED]
kdis.c@@/main/52
[REDACTED]: Merge From /vobs/rdbms/src/server/ram/index/kdis.c@@/main/51

```

Transaction: akruglik_bug-696737

```

e12700.msg@@/main/53
[REDACTED]: add message 14176
kkbl.c@@/main/29
[REDACTED]: fix for bug 696737: if performing PITR, allow STORAGE clause when parsing def
prsc.c@@/main/387
[REDACTED]: fix for bug 696737: do not restrict attributes which may be specified for a H

```

Transaction: sichandr_optimize_kokbint

```

kokb.c@@/main/16
[REDACTED]
kprb.h@@/main/14
[REDACTED]: add kprbuu* (callback bind) support for ADTs

```

Transaction: aksrivas_bug-691448

```

kge.h@@/main/41
[REDACTED]: fix bug 691448, add KGENMFASSERT macro and kgeanmfe proto
kge.c@@/main/59
[REDACTED]: fix bug 691448, add named fatal assert routine
kjga.h@@/main/7
[REDACTED]: fix bug 691448, add named fatal assert macros

```

Transaction: hasun_handle_grpby_cols_correctly_for_mavs

```

kkz.c@@/main/127
[REDACTED]: Use a unique index for MAVs instead of PK constraint
kkzg.c@@/main/28
[REDACTED]: Modify kkzgindcb() to generate unique indexes for MAVs

```

Transaction: ramkrish_ramkrish_iotlob_atbmov

```

kkm.c@@/main/379
[REDACTED]
kkpo.c@@/main/40
[REDACTED]
atb.c@@/main/227
[REDACTED]
kkb.c@@/main/82
[REDACTED]
kokl2.c@@/main/29
[REDACTED]
kditc.h@@/main/10
[REDACTED]
kdic.c@@/main/113

```


[REDACTED]
gerli.c@@/main/73

[REDACTED]
prsdef.h@@/main/35

[REDACTED]
ctc.c@@/main/221

[REDACTED]
kkbl.c@@/main/28

[REDACTED]
prsc.c@@/main/386

Transaction: mkamath_errmsg

e24280.msg@@/main/73

[REDACTED] Adding error message 25261

Transaction: sbedarka_bug-664195_1

catalog.sql@@/main/301

Transaction: sbedarka_bug-683833_1

kkdc.c@@/main/81

Transaction: sbedarka_bug-475585_1

kvpf.c@@/main/11

Transaction: lkaplan_sndcmp_delete

ColumnDescriptor.java@@/main/6

Transaction: clei_bug-696853_1

kzra.c@@/main/8

Transaction: lbarton_bug-654891

exuevw.c@@/main/25

[REDACTED] Use explicit array index in offsetof

Transaction: dalpern_snapshot_purity_adjustments_errmsg

e29250.msg@@/main/91

[REDACTED] base purity decisions on DETERMINISTIC

[REDACTED]

Transaction: gtaracha_bug-566533

opndef.h@@/main/124

[REDACTED] Removing OPTHTOR and OPTATOH

odfdef.c@@/main/82

[REDACTED]: Fixing bug 566533

koke.h@@/main/25

[REDACTED] (566533): Removing kokeehtr, kokeihtr

koke.c@@/main/39

[REDACTED] (566533): Removing kokeehtr, kokeihtr and OPTATOH from kokeithx

kkzp.c@@/main/24

[REDACTED] Removing OPTHTOR and OPTATOH

Transaction: nramani_ldap_errors

e24280.msg@@/main/74

[REDACTED] adding ldap integration errors

Out of Transaction changes:

.labellog@@/main/209

[REDACTED] new log

.manifest@@/main/solaris/216

[REDACTED] new label

ctc.c@@/main/222

[REDACTED] fix comp err

=====

Short regress has 0 dif - run with use_ism=false.

tk0irddl.suc	tkagrawt.suc	tkoolqyi.suc	tkprddls.suc	tkqrdrds.suc
tk0iriot.suc	tkb2srg.suc	tkoommo2.suc	tkprdemo.suc	tkqrsecb.suc
tk0irmts.suc	tkdrpls1.suc	tkoqbrqr.suc	tkprdepl.suc	tkqrsels.suc
tkaqbdq2.suc	tkgrsecs.suc	tkp83cl.suc	tkprdep2.suc	tkqrtres.suc
tkaqbeq1.suc	tkhodbck.suc	tkpasr81.suc	tkprdep3.suc	tkqrwhes.suc
tkaqbeq2.suc	tkhomrg.suc	tkpasrg0.suc	tkprdep4.suc	tkqxeisr.suc
tkaqdb1.suc	tkhoucmr.suc	tkpbd1.suc	tkprdep5.suc	tkrcrash.suc
tkaqddb2.suc	tkigfstt.suc	tkpmsrl.suc	tkprdep6.suc	tkrcrsha.suc
tkaqdqba.suc	tkirenbs.suc	tkpmsrnl.suc	tkprexec.suc	tkrmlite.suc
tkaqdgrc.suc	tkirinds.suc	tkpopro.suc	tkprmisc.suc	tkrmliti.suc
tkaqmbdg.suc	tkoodcat.suc	tkpotta.suc	tkprsqpl.suc	tkrsrsubs.suc
tkaqmbei.suc	tkoodcta.suc	tkpgcafs.suc	tkprtme.suc	tktrclus.suc
tkaqmnav.suc	tkoodctn.suc	tkpgdups.suc	tkprtimm.suc	tktrmnp.suc
tkaqnsbl.suc	tkoodtcl.suc	tkpggrps.suc	tkpuori.suc	tktrnaos.suc
tkaqnsb2.suc	tkoodtci.suc	tkpgjois.suc	tkqrcats.suc	tktrtabs.suc
tkaqnsb3.suc	tkoodtcm.suc	tkpgmain.suc	tkqrdats.suc	tktrvies.suc
tkaqnsb4.suc	tkoodtir.suc	tkpgords.suc	tkqrexps.suc	tkxabrch.suc
tkaqnsb5.suc	tkoodxu0.suc	tkpqrgrs.suc	tkqgrgps.suc	tkzrls00.suc
tkaqoci2.suc	tkoolqyc.suc	tkpgwhrs.suc	tkqrgjois.suc	
tkaqrawi.suc	tkoolqyd.suc	tkprbugs.suc	tkqrnuls.suc	

Long regress has been started.

Transaction: svedala_fix_cdemosyev

cdemosyev.c@@/main/st_rdbms_8.1.4/1

Transaction: gbhatia_define_net_use_ldap_flag

s_rdbms.mk@@/main/osds/unix/solaris/st_rdbms_8.1.4/1
Define NET_USE_LDAP flag

Transaction: mtakashi_merge_jox_981022

jox.c@@/main/st_rdbms_8.1.4/8
merge from cvs


Transaction: skmishra_fix-dif-4

opiodr.c@@/main/st_rdbms_8.1.4/4

Out of Transaction changes:

.depprod@@/main/solaris/st_rdbms_8.1.4/15
update javavm
.labellog@@/main/st_rdbms_8.1.4/15

```
[REDACTED] new log
.manifest@@/main/solaris/st_rdbms_8.1.4/16
[REDACTED]: new label
rdbmsgaenv@@/main/st_rdbms_8.1.4/2
[REDACTED]: Added entries to T_SOURCE, T_COM, etc. to stay in sync with .ndeprodenv
=====
```



Short regress has 0 dif - run with use_ism=false.

tk0irddl.suc	tkb2srg.suc	tkogbrqr.suc	tkprdemo.suc	tkqrsels.suc
tk0iriot.suc	tkdrpls1.suc	tkp83c1.suc	tkprdepl.suc	tkqrtres.suc
tk0irmts.suc	tkgrsecs.suc	tkpasc81.suc	tkprdep2.suc	tkqrwhes.suc
tkaqbdaq2.suc	tkhodbck.suc	tkpash81.suc	tkprdep3.suc	tkqxeisr.suc
tkaqbeq1.suc	tkhosrg.suc	tkpasrg0.suc	tkprdep4.suc	tkrcrash.suc
tkaqbeq2.suc	tkhoucsr.suc	tkpbd1.suc	tkprdep5.suc	tkrcrsha.suc
tkaqdqbl1.suc	tkigfsrt.suc	tkpdp1ipo.suc	tkprdep6.suc	tkrmlite.suc
tkaqdqgb2.suc	tkirenbs.suc	tkpmsr1.suc	tkprexec.suc	tkrmliti.suc
tkaqdqba.suc	tkirinds.suc	tkpmsrnl.suc	tkprmisc.suc	tkrsrsubs.suc
tkaqdqrc.suc	tkoodcat.suc	tkpopro.suc	tkprsqlp.suc	tktrclus.suc
tkaqmbdq.suc	tkoodcta.suc	tkpotta.suc	tkprtime.suc	tktrmnp.suc
tkaqmbei.suc	tkoodctn.suc	tkpgcafs.suc	tkprtimm.suc	tktrnaos.suc
tkaqmnav.suc	tkoodtcl.suc	tkpgdups.suc	tkpuori.suc	tktrtabs.suc
tkaqnsb1.suc	tkoodtci.suc	tkpggrps.suc	tkqrcats.suc	tktrvies.suc
tkaqnsb2.suc	tkoodtcm.suc	tkpgjois.suc	tkqrdsats.suc	tkkxbrch.suc
tkaqnsb3.suc	tkoodtir.suc	tkpqmain.suc	tkqrexps.suc	tkzrlsd0.suc
tkaqnsb4.suc	tkoodxu0.suc	tkpgords.suc	tkqgrgps.suc	tkzrlso0.suc
tkaqnsb5.suc	tkoolqyc.suc	tkpqrgrs.suc	tkqgrjois.suc	
tkaqoci2.suc	tkoolqyd.suc	tkpqwhrs.suc	tkqqrnl.suc	
tkaqrawi.suc	tkoolqyi.suc	tkprbugs.suc	tkqqrords.suc	
tkaqrawt.suc	tkoommo2.suc	tkprddls.suc	tkqrsecb.suc	

Long regress has been started.

Transaction: nvishnub_bug-786151

exuept.c@@/main/14

Transaction: gtarora_kollasg_OCIObjSetAttr

orid.c@@/main/8
lint

Transaction: anithrak_fix_state_obj_dump

kcb.h@@/main/167
Add new where for kcbmb1 for call from kcbzps
kcbz.c@@/main/149

Transaction: whe_fix_lrg31076

orl.c@@/main/18

[REDACTED]: lrg31076: typo in OCIOpaqueCtxGetHandles

Transaction: jfeenan_merger

gsmgnt1.c@@/main/13

[REDACTED]: reduce invalidation and lock window

Transaction: fge_bug-705269

qkadv.c@@/main/256

[REDACTED] # (705269): qkadv projects rwo for bitmap dn if ROWNUM

qkna.c@@/main/25

[REDACTED] # (705269): qknpxAllocate: new rwo if bitmap dn underneath

Transaction: gbhatia_ano_radius_encrypt_passwd

kpuzln.c@@/main/31

[REDACTED]: ANO Radius Fix

upilog.c@@/main/63

[REDACTED]: ANO Radius Fix

kzia.c@@/main/14

[REDACTED]: ANO Radius Fix

Out of Transaction changes:

.depprod@@/main/solaris/293

[REDACTED]: update plsqli, precomp, sqlplus, ordts, oemagent

.labellog@@/main/334

[REDACTED]: new log

.manifest@@/main/solaris/341

[REDACTED]: new label

=====